

On Sparser Random 3SAT Refutation Algorithms and Feasible Interpolation*

Iddo Tzameret[†]

Abstract

We formalize a combinatorial principle, called *the 3XOR principle*, due to Feige, Kim and Ofek [FKO06], as a family of unsatisfiable propositional formulas for which refutations of small size in any propositional proof system that possesses the feasible interpolation property imply an efficient *deterministic* refutation algorithm for random 3SAT with n variables and $\Omega(n^{1.4})$ clauses. Such small size refutations would improve the current best (with respect to the clause density) efficient refutation algorithm, which works only for $\Omega(n^{1.5})$ many clauses [FO07].

We then study the proof complexity of the above formulas in weak extensions of cutting planes and resolution. Specifically, we show that there are polynomial-size refutations of the 3XOR principle in resolution operating with disjunctions of quadratic equations (with small integer coefficients), denoted $R(\text{quad})$. We show that $R(\text{quad})$ is weakly automatizable iff $R(\text{lin})$ is weakly automatizable, where $R(\text{lin})$ is similar to $R(\text{quad})$ but with linear instead of quadratic equations (introduced in [RT08]). This reduces the question of the existence of efficient deterministic refutation algorithms for random 3SAT with n variables and $\Omega(n^{1.4})$ clauses to the question of feasible interpolation of $R(\text{quad})$ and to the weak automatizability of $R(\text{lin})$.

1 Introduction

In the well known *random 3SAT model* one usually considers a distribution on formulas in conjunctive normal form (CNF) with m clauses and three literals each, where each clause is chosen independently with repetitions out of all possible $2^3 \cdot \binom{n}{3}$ clauses with n variables. The *clause density* of such a 3CNF is m/n . When m is greater than cn for sufficiently large c , that is, when the clause density is greater than c , it is known (and easily proved for $c \geq 5.2$) that with high probability a random 3CNF is unsatisfiable.

A *refutation algorithm* for random k CNFs is an algorithm that receives a k CNF (with c sufficiently large) and outputs either “unsatisfiable” or “don’t know”; if the algorithm answers “unsatisfiable” then the k CNF is required to be indeed unsatisfiable; moreover, the algorithm should output “unsatisfiable” with high probability (namely, with probability $1 - o(n)$ over the input k CNFs).

*Supported in part by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, the National Natural Science Foundation of P. R. China; Grant 61033001, 61061130540, 61073174.

[†]Institute for Theoretical Computer Science, The Institute for Interdisciplinary Information Sciences (IIIS), Tsinghua University, Beijing tzameret@tsinghua.edu.cn

Refutation algorithms for random k CNFs were investigated in Goerdts and Krivelevich [GK01] and subsequent works by Goerdts and Lanka [GL03], Friedman, Goerdts and Krivelevich [FGK05], Feige and Ofek [FO07] and Feige [Fei07] (among other works). The best (with respect to the clause density) refutation algorithm to date works for random 3CNFs with at least $\Omega(n^{1.5})$ clauses [FO07]. On the other hand, Feige, Kim and Ofek [FKO06] considered efficient *non-deterministic* refutation algorithms (in other words, short *witnesses* for unsatisfiability of 3CNFs that can be checked for correctness in polynomial-time). They established the current best (again, with respect to the clause density) efficient, alas *non-deterministic*, refutation procedure: they showed that with probability converging to 1 a random 3CNF with n variables and $\Omega(n^{1.4})$ clauses has a polynomial-size (in n) witness. It is an open problem to devise a *deterministic* polynomial-time (or even a quasipolynomial-time) refutation algorithm for random 3CNFs with n variables and $\Omega(n^{1.4})$ clauses.

In this work we reduce the problem of devising an efficient deterministic refutation algorithm for random 3CNFs with $\Omega(n^{1.4})$ clauses to a problem in propositional proof complexity, namely the feasible interpolation problem. Informally, the feasible interpolation property gives a way to transform short refutations of an unsatisfiable formula $A(x, y) \wedge B(x, z)$, for x, y, z pairwise disjoint sets of variables, into small circuits that given an input α , separate the cases where $A(\alpha, y)$ is satisfiable from the cases where $B(\alpha, z)$ is satisfiable (see Definition 1). Feasible interpolation was proposed in [Kra94] and developed further in [Raz95, BPR97, Kra97]. We present a family of unsatisfiable propositional formulas **denoted** Υ_n expressing a combinatorial principle whose short refutations in any propositional proof system that possesses the feasible interpolation property imply an efficient deterministic refutation algorithm for random 3CNFs with $\Omega(n^{1.4})$ clauses:

Theorem 1 *If there exists a propositional proof system that has the feasible interpolation property and that admits uniform polynomial-size refutations of Υ_n , then there is a deterministic polynomial-time refutation algorithm for random 3CNF formulas with $\Omega(n^{1.4})$ clauses.*¹

This implies that determinizing the Feige et al. polynomial-time non-deterministic algorithm reduces to establishing the feasible interpolation property for those propositional proof systems that admit short refutations of the corresponding formulas Υ_n (or to the problem of demonstrating polynomial-size proofs of Υ_n in a proof system whose feasible interpolation property is already established).

The argument is based on the following simple observation: we observe that the *computationally hard part* of the Feige, Kim and Ofek non-deterministic refutation algorithm (namely, the part we do not know how to efficiently compute deterministically) corresponds to a disjoint **NP**-pair (formally, it is a family of such pairs). The pair of disjoint **NP** sets are, first, the set of 3CNFs that have a certain combinatorial property, namely they contain a collection of sufficiently many *inconsistent even k -tuples*, as defined by Feige et al., and second, the set of 3CNFs with m clauses for which there exists an assignment that satisfies more than $m - \ell$ clauses as 3XORs (for ℓ a certain function of the number of variables n).

It is worth noting that any efficient refutation algorithm (deterministic or not) corresponds to a disjoint **NP**-pair, via a sort of *reflection principle*. The reason is as follows: any refutation

¹This theorem also holds if we consider feasible interpolation to be in quasipolynomial-size (which would then entail a corresponding quasipolynomial time refutation algorithm).

algorithm is based on some property P of CNFs that can be witnessed (or better, found) in polynomial-time. Thus, any refutation algorithm corresponds to a family of formulas $P(x) \rightarrow \neg \text{SAT}(x)$, expressing that if the input CNF has property P then x is unsatisfiable, and so $P(x)$ and $\text{SAT}(x)$ are two disjoint **NP** predicates. However, the disjoint **NP**-pair we work with is *not* of this type. That is, $P(x)$ will not be the predicate for the full Feige, Kim and Ofek witnesses, rather a specific (combinatorial) predicate, that is only one ingredient in the definition of the Feige et al. witness. This saves us the trouble to formalize and prove in a weak propositional proof system the full Feige et al. argument (such a full formalization was done inside TC^0 -Frege in [MT12]). For more information on the relation between disjoint **NP**-pairs and propositional proof complexity see for example [Pud03, AB04].

In the second part of this paper, we study the above suggested approach for improving the current known refutation algorithms, by considering short refutations for Υ_n in seemingly weak proof systems that extend both the cutting planes system (cf. [CCT87, Pud97]) and $\text{Res}(2)$ (the system $\text{Res}(k)$ is an extension of resolution that operates with k DNFs instead of clauses, introduced by Krajíček [Kra01]). Note that the cutting planes proof system itself has feasible interpolation [BPR97, Kra97, Pud97]. But on the other hand, it is unknown whether $\text{Res}(2)$ has feasible interpolation, while it is known by Atserias and Bonnet [AB04] that $\text{Res}(2)$ has feasible interpolation iff resolution is weakly automatizable (see below on weak automatizability).

More precisely, a family of $\text{R}(\text{quad})$ refutations is a family of refutations operating with *disjunctions* of quadratic equations, where each quadratic equation is of the form:

$$\sum_{i,j \in [n]} c_{ij} x_i x_j + \sum_{i \in [n]} c_i x_i + c_0 = a$$

such that c_i, c_{ij} and a are integers written in unary representation. We have the following (generalized) resolution rule: from two disjunctions of quadratic equations $\bigvee_i L_i \vee L = a$ and $\bigvee_j L_j \vee L' = b$ we can derive:

$$\bigvee_i L_i \vee \bigvee_j L_j \vee (L - L' = a - b).$$

We also need to add axioms that will force our variables to be 0,1 (see Section 5.1).

We show the following:

Theorem 2 $\text{R}(\text{quad})$ admits polynomial-size refutations of the 3XOR principle Υ_n .

This means that proving that $\text{R}(\text{quad})$ has the feasible interpolation (in polynomial-size or quasi-polynomial-size) would entail a deterministic (polynomial-time or quasipolynomial-time, respectively) refutation algorithm for random 3CNFs with $\Omega(n^{1.4})$ clauses.

Further, note that [MT12] proved the 3XOR principle in TC^0 -Frege, which we might conjecture to be a stronger proof system than $\text{R}(\text{quad})$. Thus, Theorem 2 gives a possibly tighter logical characterization of the 3XOR principle than [MT12]. On the other hand, [MT12] gave a polynomial-size proof for the correctness of the full Feige et al. witnesses (and not only the 3XOR principle).

To describe our next result we recall the concept of automatizability which is central to proof-search algorithms. Given any propositional proof system, an important question

is whether one can *find* efficiently proofs in the system (regardless of their sizes). For this purpose the concept of automatizability was introduced by Bonet et al. [BPR00]: we say that a refutation system \mathcal{P} (equivalently, a proof-system) is *automatizable* if there exists a polynomial-time algorithm that on input $(\tau, 1^m)$, where τ is an unsatisfiable formula and 1^m is a string of m ones, the algorithm outputs a \mathcal{P} -refutation of τ of size at most m in case such a refutation exists. Following Atserias and Bonet [AB04], we say that a refutation system \mathcal{P} is *weakly automatizable* if there exists an automatizable refutation system \mathcal{P}' that polynomially simulates \mathcal{P} . Note that if \mathcal{P} is not automatizable, it does not imply that \mathcal{P}' is not automatizable. Hence, from the perspective of proof-search algorithms, weak automatizability is a more natural notion than automatizability (see [Pud03] on this).

In [RT08], the system $R(\text{lin})$ was introduced which is similar to $R(\text{quad})$ above, except that all equations are linear instead of quadratic. Using similar arguments to Pudlák [Pud03], we show the following:

Theorem 3 *$R(\text{quad})$ is weakly automatizable iff $R(\text{lin})$ is weakly automatizable.*

Since weak automatizability of a proof system implies also that the proof system has feasible interpolation, we obtain the following reduction:

Corollary 4 *If $R(\text{lin})$ is weakly automatizable then there is a deterministic refutation algorithm for random 3CNFs with $\Omega(n^{1.4})$ clauses.*

1.0.1 Discussion

The key point of this work is the reduction from constructing an efficient refutation algorithm for the clause density $\Omega(n^{0.4})$ to proving upper bounds in weak enough propositional proof systems for the 3XOR principle. On the one hand, this might be construed as hinting on a new way to improve current refutation algorithms. Note that even establishing a quasipolynomial interpolation for proof systems that admit short proofs of the 3XOR principle would be very interesting since the best known refutation algorithm for $\Omega(n^{1.4})$ clauses works in time $2^{O(n^{0.2} \log n)}$ [FKO06]. However, it is probably safer to consider this reduction as emphasizing the important consequences that establishing the feasible interpolation property for $\text{Res}(2)$ and its (weak) extensions would have. For more recent work on the algorithmic implications of establishing feasible interpolation and automatizability for related propositional proof systems see [AM12, HP11].

The rationale behind choosing the $R(\text{quad})$ refutation system is that it seems to be a possibly weak propositional proof system that, loosely speaking, can both *count* and *compose mappings*, as we now explain. The cutting planes proof system can count to a certain extent; however, it is unknown whether cutting planes has short refutations of the Tseitin graph formulas [Tse68] (which are a kind of counting contradictions). By extending cutting planes (with small coefficients) to allow disjunctions of linear equations (instead of a single inequality), we obtain the system $R(\text{lin})$ (this is similar to $R(\text{quad})$ but with linear instead of quadratic equations). It was shown in [RT08] that even when we allow disjunctions of only *constant many generalized*² linear equations in every proof-line, $R(\text{lin})$ has short refuta-

²A *generalized equation* is an equation $L \in S$, for $S \subset \mathbb{Z}$; which can be considered also as the disjunction $\bigvee_{s \in S} L = s$.

tions of the Tseitin formulas; this shows that using (fairly restricted) disjunctions of linear equations allows to improve the counting abilities of cutting planes.

The reason we need to use quadratic instead of linear equations is to be able to compose maps. As observed by Pudlák [Pud03], the reason why the k -Clique and $(k-1)$ -Coloring contradictions do not have short cutting planes refutations is that cutting planes cannot compose two mappings, which then makes it impossible to perform a routine reduction from the k -Clique and $(k-1)$ -Coloring contradiction to the pigeonhole principle contradiction (and the latter contradiction does admit short cutting planes refutations). This is why Pudlák introduced in [Pud03] the system CP^2 which is cutting planes operating with *quadratic equations*. The system $R(\text{quad})$ we work with is an extension of CP^2 (when the latter is restricted to small coefficients).

2 Preliminaries

Let F be a 3CNF with n variables $X = \{x_1, \dots, x_n\}$ and m clauses. We denote $\{1, \dots, n\}$ by $[n]$. The truth value of a formula G under the Boolean assignment A is written $G(A)$. An assignment A *satisfies as a 3XOR* the clause $\ell_1 \vee \ell_2 \vee \ell_3$ if $(\ell_1 \oplus \ell_2 \oplus \ell_3)(A) = 1$ (where \oplus denotes the XOR operation, and the ℓ_i 's are literals, namely variables or their negation).

2.1 Disjoint NP-pairs and feasible interpolation of propositional proofs

In this section we review the notion of a disjoint **NP**-pair and its relation to propositional proofs and the feasible interpolation property.

A *disjoint NP-pair* is simply a pair of languages in **NP** that are disjoint. Let L, N be a disjoint **NP**-pair such that $R(x, y)$ is the corresponding relation for L and $Q(x, z)$ is the corresponding relation for N ; namely, $R(x, y)$ and $Q(x, z)$ are polynomial-time relations such that $x \in L$ iff $\exists y, |y| = \text{poly}(|x|) \wedge R(x, y) = \text{true}$ and $x \in N$ iff $\exists z, |z| = \text{poly}(|x|) \wedge Q(x, z) = \text{true}$. Then, for every $n \in \mathbb{N}$, there exists an *unsatisfiable* CNF formula in three mutually disjoint sets of variables $\bar{x}, \bar{y}, \bar{z}$:

$$F_n := A(\bar{x}, \bar{y}) \wedge B(\bar{x}, \bar{z}), \quad (1)$$

where $A(\bar{x}, \bar{y})$ and $B(\bar{x}, \bar{z})$ are the CNF formulas expressing that $R(x, y)$ and $Q(x, z)$ are true for x of length n , respectively.

More formally, $A(\bar{x}, \bar{y})$ is a CNF in the variables $\bar{x} = (x_1, \dots, x_n)$ and $\bar{y} = (y_1, \dots, y_\ell)$, that is true iff $R(\bar{x}, \bar{y})$ is true, and $B(\bar{x}, \bar{z})$ is a CNF in the variables \bar{x} and $\bar{z} = (z_1, \dots, z_m)$, that is true iff $Q(\bar{x}, \bar{z})$ is true (for some ℓ, m that are polynomial in n). Since both polynomial-time relations $R(x, y)$ and $Q(x, z)$ can be converted into a family of polynomial-size Boolean circuits, and thus can be written as a family of polynomial-size CNF formulas (by adding extension variables, that we may assume are incorporated in the certificates \bar{y} and \bar{z}), both the formulas $A(\bar{x}, \bar{y})$ and $B(\bar{x}, \bar{z})$ can be written as polynomial-size CNF formulas.

Note that because \bar{y} and \bar{z} are disjoint sets of variables and $A(\bar{x}, \bar{y}) \wedge B(\bar{x}, \bar{z})$ is unsatisfiable, it must be that given any $\bar{x} \in \{0, 1\}^n$, either $A(\bar{x}, \bar{y})$ or $B(\bar{x}, \bar{z})$ is unsatisfiable (or both).

Feasible Interpolation. We use standard notation and notions from the theory of propositional proof complexity (see [BP98, Seg07, CK02, Kra95] for surveys and introductions to the

field). In particular, we sometimes mix between refutations (that is, proofs of unsatisfiability of a formula) and proofs (that is, a proof of the negation of an unsatisfiable formula, which must be a tautology). From the perspective of proof complexity refutations of contradictions and proofs of tautologies are mostly the same.

Definition 1 (Interpolation property) *A propositional proof system \mathcal{P} is said to have the interpolation property in size $f(n)$ if the existence of a \mathcal{P} -refutation of $A(\bar{x}, \bar{y}) \wedge B(\bar{x}, \bar{z})$ as in (1) of size s implies the existence of (usually a Boolean) circuit $C(\bar{x})$ of size $f(s)$, called the interpolant of $A(\bar{x}, \bar{y}) \wedge B(\bar{x}, \bar{z})$, such that*

$$\begin{aligned} C(\bar{\alpha}) = 1 &\implies A(\bar{\alpha}, \bar{y}) \text{ is unsatisfiable; and} \\ C(\bar{\alpha}) = 0 &\implies B(\bar{\alpha}, \bar{z}) \text{ is unsatisfiable.} \end{aligned} \tag{2}$$

In other words, if only $A(\bar{\alpha}, \bar{y})$ is unsatisfiable then $C(\bar{\alpha}) = 1$ and if only $B(\bar{\alpha}, \bar{z})$ is unsatisfiable then $C(\bar{\alpha}) = 0$, and if both $A(\bar{\alpha}, \bar{y})$ and $B(\bar{\alpha}, \bar{z})$ are unsatisfiable then $C(\bar{\alpha})$ can output either 0 or 1. When a proof system \mathcal{P} has the interpolation property in size $\text{poly}(n)$ we say that \mathcal{P} has the feasible interpolation property, or simply that \mathcal{P} has feasible interpolation.

If the \mathcal{P} -refutations of the family of formulas F_n (in (1) above) are uniform, in the sense that there is a polynomial-time algorithm that can construct the refutations of F_n given n in unary representation, then the interpolant circuit C of F_n above is assumed to be uniform as well (and so if \mathcal{P} has feasible interpolation then there is a polynomial-time algorithm that behaves as the circuit C in (2)).

Note that L (as defined above) is precisely the set of those assignments $\bar{\alpha}$ for which $A(\bar{\alpha}, \bar{y})$ is satisfiable, and N is precisely the set of those assignments $\bar{\alpha}$ for which $B(\bar{\alpha}, \bar{z})$ is satisfiable, and L and N are disjoint by assumption, and so $C(\bar{x})$ separates L from N ; namely, it outputs different values for those elements in L and those elements in N .

It is already known that quite a few propositional proof systems possess the feasible interpolation property. Such systems include for instance resolution [BPR97, Pud97, Kra97], cutting planes [IPU94, BPR97, Pud97, HC99] and the polynomial calculus [CEI96]. On the other hand it is known (conditioned on hardness assumptions from complexity theory) that several other proof systems do not have the feasible interpolation property; these include threshold logic (namely, TC^0 -Frege) [BPR00] and bounded depth Frege [BDG⁺99]

3 The 3XOR principle

The following definitions and proposition is due to Feige et al. [FKO06].

Definition 2 (Inconsistent even k -tuple) *An even k -tuple is a tuple of k many 3-clauses in which every variable appears even times. An inconsistent even k -tuple is an even k -tuple in which the total number of negative literals is odd.*

Note that for any even k -tuple, k must be an even number (since by assumption the total number of variables occurrences $3k$ is even). The following is the combinatorial principle, due to Feige et al. [FKO06] that we consider in this work:

The 3XOR principle: Let K be a 3CNF over the variables X . Let S be t inconsistent even k -tuples from K , such that every clause from K appears in at most d inconsistent even k -tuples in S . Then, given any Boolean assignment to the variables X , the number of clauses in K that are unsatisfied by the assignment as 3XOR is at least $\lceil t/d \rceil$.

The correctness of the 3XOR principle follows directly from the following proposition:

Proposition 5 ([FKO06]) *For any inconsistent even k -tuple (over the variables X) and any Boolean assignment A to X , there must be a clause in the k -tuple that is unsatisfied as 3XOR.*

Proof: Assume by a way of contradiction that for some assignment A every clause from the k -tuple is satisfied as a 3XOR. Remember that k must be even. Thus, if we sum modulo 2 all the literals in the k -tuple *via clauses*, then since k is even we get that the sum equals 0 modulo 2.

On the other hand, if we count *via literals* then summing modulo 2 all literals $\ell_i(A)$ in the k -tuple, we get 1 (modulo 2), for the following reason. First, we sum all variables x_i that have odd number of negative occurrences. Because x_i appears an even number of times in the k -tuple, the number of positive occurrences of x_i is also odd. So in total all occurrences of $x_i(A)$ and $\neg x_i(A)$ contribute 1 to our sum (modulo 2). There must be an odd number of such variables x_i in our k -tuple because the k -tuple is *inconsistent*. Thus this sums up to 1 (modulo 2). Then we add to this sum those variables that have an even number of negative occurrences (and hence also an even number of positive occurrences); but they cancel out when summing their values under A modulo 2, and so they contribute 0 to the total sum. Hence, we get 1 as the total sum. This contradicts the counting in the previous paragraph which turned out 0. ■

Note that the 3XOR principle stems directly from the Proposition 5 and the fact that every clause in K appears in at most d even k -tuples in S .

4 From short proofs to refutation algorithms

In this section we demonstrate that polynomial-size proofs of (an encoding of the) 3XOR principle in a proof system that has the feasible interpolation property yield deterministic polynomial-time refutation algorithms for random 3CNF formulas with $\Omega(n^{1.4})$ clauses.

4.1 The witness for unsatisfiability

Feige, Kim and Ofek nondeterministic refutation algorithm [FKO06] is based on the existence of a polynomial-size witness of unsatisfiability for most 3CNF formulas with sufficiently large clause to variable ratio. The witness has several parts, but as already observed in [FKO06], apart from the t inconsistent even k -tuples (Definition 2), all the other parts of the witness are known to be computable in polynomial-time. In what follows we define the witnesses for unsatisfiability.

Let K be a 3CNF with n variables x_1, \dots, x_n and m clauses. The *imbalance* of a variable x_i is the absolute value of the difference between the number of its positive occurrences and the number of its negative occurrences. The *imbalance of K* is the sum over the imbalances

of all variables, denoted $I(K)$. We define $M(K)$ to be an $n \times n$ rational matrix such that M_{ij} equals $\frac{1}{2}$ times the number of clauses in K where x_i and x_j appear with different signs minus $\frac{1}{2}$ times the number of clauses where they appear with the same sign. In other words, for each clause in K in which x_i and x_j appear with the same sign we add $\frac{1}{2}$ and for each clause in K in which x_i and x_j appear with different signs we subtract $\frac{1}{2}$. Let λ be a rational approximation of the biggest eigenvalue of $M(K)$. (It is enough to use polynomially small rational approximations; this was already noted in [FKO06]; see also [MT12] for a detailed treatment of rational approximations in this context.)

Definition 3 (FKO witness) *Given a 3CNF K , the FKO witness for the unsatisfiability of K is defined to be the following collection:*

1. *the imbalance $I(K)$;*
2. *the matrix $M(K)$ and the (polynomially small) rational approximation λ of its largest eigenvalue;*
3. *a collection S consisting of $t < n^2$ inconsistent even k -tuples such that every clause in K appears in at most d many even k -tuples;*
4. *the inequality $t > \frac{d \cdot (I(K) + \lambda n)}{2} + o(1)$ must hold.*

(The notation $o(1)$ above stands for a specific rational number b/n^c , for c a constant and b a positive integer.)

We have the following:

Theorem 6 ([FKO06]) *For a random 3CNF K with n variables and $\Omega(n^{1.4})$ clauses, with probability converging to 1 as n tends to infinity there exist natural numbers $k = O(n^{0.2})$, $t = \Omega(n^{1.4})$ (where $t < n^2$) and $d = O(k) = O(n^{0.2})$, such that K has a witness for unsatisfiability as in Definition 3.*

It can be shown that given a 3CNF K and the parameter k , we can compute in polynomial time the values of t and d , for which the theorem holds.

Inspecting the argument in [FKO06], one can see that it is sufficient to replace part 3 in the witness with a witness for the following:

- 3'. *No assignment can satisfy more than $m - \lceil t/d \rceil$ clauses in K as 3XORs.*

Therefore, since $I(K)$, $M(K)$ and λ are all polynomial-time computable (see [FKO06] for this), in order to determinize the non-deterministic refutation algorithm of [FKO06] it is sufficient to provide an algorithm that almost surely determines (correctly) that part 3' above holds. In other words, in order to construct an efficient refutation algorithm for random 3CNFs (with $\Omega(n^{1.4})$ clauses) it is sufficient to have a deterministic algorithm A that on every input 3CNF answers either “condition 3' is correct” or “don't know”, such that A is never wrong (i.e., if it says “condition 3' is correct” then condition 3' holds) and with probability $1 - o(n)$ over the input 3CNFs most inputs A answers “condition 3' is correct”. Note that we do not need to actually find the Feige et al. witness nor do we need to decide if it exists or not. The relation between unsatisfiability and bounding the number of clauses that can be satisfied as 3XOR in a 3CNF was introduced by Feige in [Fei02] (and used in [FO07] as well as in [FKO06]).

4.2 The disjoint NP-pair corresponding to the 3XOR principle

For every choice of t, d and k functions of n (that are polynomially in n computable; we assume also that $t(n) \leq n^2$), we define the corresponding *3XOR principle disjoint NP-pair* as the pair (L, N) , where:

$$L := \{x \mid x \text{ is a 3CNF with } n \text{ variables and } m \text{ clauses for which} \\ \text{there exists } t(n) \text{ inconsistent even } k(n)\text{-tuples, and} \\ \text{every clause of } x \text{ appears in at most } d(n) \text{ } k(n)\text{-tuples}\},$$

$$N := \{x \mid x \text{ is a 3CNF with } n \text{ variables and } m \text{ clauses for which} \\ \text{there exists an assignment that satisfies at least} \\ m - (I(x) + \lambda n)/2 \text{ clauses as 3XOR}\},$$

where $I(x)$ in the definition of N is the imbalance of x and λ is the (polynomially small) rational approximation of the largest eigenvalue of $M(x)$ as described in Section 4.1.

Formally, for every choice of t, k, d as functions of n there is a different corresponding disjoint **NP**-pair. However, for the sake convenience we do not specify explicitly the chosen t, k, d in the notation L, N as it will be clear from the context which t, k, d we use. It is easy to verify that both L and N are indeed **NP** sets for every choice of $t < n^2, d, k, m$, and that, by the 3XOR Principle, $L \cap N = \emptyset$.

Using the same notation as in Section 2.1, we denote by $R(x, y)$ and $Q(x, z)$ the polynomial-time relations for L and N , respectively. Further, for every $n \in \mathbb{N}$, there exists an *unsatisfiable* CNF formula in three mutually disjoint sets of variables $\bar{x}, \bar{y}, \bar{z}$:

$$\Upsilon_n := A(\bar{x}, \bar{y}) \wedge B(\bar{x}, \bar{z}), \quad (3)$$

where $A(\bar{x}, \bar{y})$ and $B(\bar{x}, \bar{z})$ are the CNF formulas expressing that $R(x, y)$ and $Q(x, z)$ are true for x of length n , respectively.

Corollary 7 *If there exists a propositional proof system that possesses the feasible interpolation property and that admits uniform polynomial-size refutations of Υ_n , then there is a deterministic polynomial-time refutation algorithm for random 3CNF formulas with $\Omega(n^{1.4})$ clauses.*

Note: For the consequence in the corollary to hold it is enough to assume that there are polynomial-size refutations of Υ_n where the parameters are set as follows: $k = O(n^{0.2})$, $t = \Omega(n^{1.4})$ (and also $t < n^2$), $d = O(k) = O(n^{0.2})$; however, for the short proofs in the next section we only need to assume that t is polynomial in n .

Proof: By assumption, and by the definition of the feasible interpolation property, for every choice of functions t, d, k of n (that are also polynomially in n computable) there exists a deterministic polynomial-time interpolant algorithm A that on input a 3CNF K , if $A(K) = 1$ then $K \notin L$ and if $A(K) = 0$ then $K \notin N$.

The desired refutation algorithm then works as follows: it receives the 3CNF K and for all polynomially many choices of $t < n^2, d, k$ it runs $A(K)$. If for one of these runs $A(K) = 0$

then we know that $K \notin N$, and we answer “unsatisfiable”. Otherwise, we answer “don’t know”.

The correctness of this algorithm stems from the following two points:

- (i) If we answered “unsatisfiable”, K is indeed unsatisfiable, since $K \notin N$, and by Section 4.1 this means that K is unsatisfiable.
- (ii) For almost all 3CNFs we will answer “unsatisfiable” because almost all of them will have an FKO witness (by Theorem 6), which means that $K \in L$ for some choice of $t < n^2, d, k$ and hence the interpolant algorithm **A** must output 0 in at least one of these cases (because $A(K) = 1$ means that $K \notin L$). ■

5 Propositional formula encoding the 3XOR principle

In this section we define the propositional refutation system in which we are going to demonstrate polynomial-size refutations of the 3XOR principle. We then give an explicit encoding of the 3XOR principle as an unsatisfiable set of disjunctions of linear equations.

5.1 The propositional refutation systems $R(\text{lin})$ and $R(\text{quad})$

The refutation system in which we shall prove the unsatisfiability of the 3XOR principle is denoted $R(\text{quad})$. It is an extension of the refutation system $R(\text{lin})$ introduced in [RT08]. The system $R(\text{lin})$ operates with disjunctions of linear equations with integer coefficients and $R(\text{quad})$ operates with disjunctions of quadratic equations with integer coefficients, where in both cases the coefficients are written in unary representation. We also add axioms that force all variables to be 0, 1. First we define the refutation system $R(\text{lin})$.

The **size** of a linear equation $a_1x_1 + \dots + a_nx_n + a_{n+1} = a_0$ is defined to be $\sum_{i=0}^{n+1} |a_i|$, that is, the sum of the bit sizes of all a_i written in *unary* notation. The *size of a disjunction of linear equations* is the total size of all linear equations in it. The **size** of a *quadratic equation* and of a disjunction of quadratic equations is defined in a similar manner (now counting the size of the constant coefficients, the coefficients of the linear terms and the coefficients of the quadratic terms). The *empty disjunction* is unsatisfiable and stands for the truth value **false**.

Notation: For L a linear or quadratic sum and $S \subseteq \mathbb{Z}$, we write $L \in S$, to denote the disjunction $\bigvee_{s \in S} L = s$. We call $L \in S$ a *generalized linear (or quadratic) equation*.

Definition 4 ($R(\text{lin})$) *Let $K := \{K_1, \dots, K_m\}$ be a collection of disjunctions of linear equations in the variables x_1, \dots, x_n . An $R(\text{lin})$ -proof from K of a disjunction of linear equations D is a finite sequence $\pi = (D_1, \dots, D_\ell)$ of disjunctions of linear equations, such that $D_\ell = D$ and for every $i \in [\ell]$ one of the following holds:*

1. $D_i = K_j$, for some $j \in [m]$;
2. D_i is a **Boolean axiom** $x_t \in \{0, 1\}$, for some $t \in [n]$;
3. D_i was deduced by one of the following $R(\text{lin})$ -inference rules, using D_j, D_k for some $j, k < i$:

Resolution Let A, B be two, possibly empty, disjunctions of linear equations and let L_1, L_2 be two linear equations. From $A \vee L_1$ and $B \vee L_2$ derive $A \vee B \vee (L_1 - L_2)$. (We assume that every linear form with n variables is written as a sum of at most $n + 1$ monomials.³)

Weakening From a possibly empty disjunction of linear equations A derive $A \vee L$, where L is an arbitrary linear equation over the variables x_1, \dots, x_n .

Simplification From $A \vee (0 = k)$ derive A , where A is a possibly empty disjunction of linear equations and $k \neq 0$.

An $R(\text{lin})$ refutation of a collection of disjunctions of linear equations K is a proof of the empty disjunction from K . The **size** of an $R(\text{lin})$ proof π is the total size of all the disjunctions of linear equations in π (where coefficients are written in unary representation).

Definition 5 ($R(\text{quad})$) The system $R(\text{quad})$ is similar to $R(\text{lin})$ except that proof-lines can be disjunctions of quadratic equations with integer coefficients $\sum_{i,j} c_{ij}x_i x_j + \sum_i c_i x_i + c = S$ instead of linear equations; and the **Boolean axioms** are now defined for all $i, j \in [n]$, as follows:

$$x_i \in \{0, 1\}, \quad x_i + x_j - x_i x_j \in \{0, 1\}, \quad x_i - x_i x_j \in \{0, 1\}.$$

The size of an $R(\text{quad})$ refutation is the total size of all the proof-lines in it.

Both $R(\text{lin})$ and $R(\text{quad})$ can be proved to be sound and complete (for their respective languages, namely, disjunctions of linear and quadratic equations, respectively) refutation systems.

5.2 The formula

We now describe the formula Υ_n encoding the 3XOR principle (the formula depends also on the parameters t, m and k , but we shall suppress these subscripts). The formula will have the correct form for application of the feasible interpolation property, namely it will be an unsatisfiable set of formulas consisting of two groups: the first will be the formulas in the $\overline{X}, \overline{Y}$ variables only and the second will be the formulas in the $\overline{X}, \overline{Z}$ variables only. We also have variables that encode a product of two variables, namely, (extension) variables for which a certain formula in one of the groups forces them to behave like products of two variables from $\overline{X}, \overline{Y}, \overline{Z}$. Since we cannot use the \overline{Y} variables in the second part of the formula and we cannot use the \overline{Z} variables in the first part of the formula, we can encode only products of variables from $\overline{X}, \overline{Y}$ and from $\overline{X}, \overline{Z}$, but *not* products of a \overline{Y} variable with a \overline{Z} variable.

It will be convenient sometimes to denote by x_{i+n} the literal $\neg x_i$, when it is assumed we use the n variables x_1, \dots, x_n in the 3CNF encoded by \overline{X} .

Variables and their meaning. The variables \overline{X} correspond to the input 3CNF with n variables. The variables \overline{Y} correspond to the collections of t many inconsistent even k -tuples. The $\overline{Z} = \{z_1, \dots, z_n\}$ variables stand for a Boolean assignment for the n variables of the 3CNF.

³Accordingly, in $R(\text{quad})$ we assume that every quadratic sum with n variables is written as a sum of at most $1 + 2n + \binom{n}{2}$ monomials.

(Note that we use the variables x_i for the variables in the 3CNF and the variables x_{ij} for the variables in our encoding of the 3CNF.)

The input 3CNF \overline{X} is encoded as a $3m \times 2n$ table \overline{X} , where each block of three rows corresponds to a clause and columns from 1 to n correspond to positive literal occurrences and columns $n + 1$ to $2n$ correspond to negative literal occurrence. Formally, let $1 \leq i = 3 \cdot l + r \leq 3m$, where $r \in \{0, 1, 2\}$, $l \in [n]$, and let $j \in [2n]$. Then $x_{ij} = 1$ means that the r th literal in the l th clause in the input 3CNF is:

$$x_j \text{ if } j \leq n, \text{ and } \neg x_{j-n}, \text{ if } j > n.$$

The collection of t inconsistent k even tuples is encoded as t tables, each table is encoded by the variables $\overline{Y}^{(s)}$, for $s \in [t]$. Each $\overline{Y}^{(s)}$ represents a table of dimension $k \times m$, where $y_{jl}^{(s)} = 1$ iff the j th member in the s th k -tuple is the l th clause (meaning the l th clause in the input 3CNF encoded by \overline{X}).

Group I of formulas (containing only $\overline{X}, \overline{Y}$):

1. Every row in \overline{X} contains exactly one 1:

$$\sum_{j=1}^{2n} x_{ij} = 1, \quad \text{for every } i \in [3m].$$

2. Every row in $\overline{Y}^{(s)}$ contains exactly one 1:

$$\sum_{j=1}^m y_{ij}^{(s)} = 1, \quad \text{for all } s \in [t], i \in [k].$$

3. Every column in $\overline{Y}^{(s)}$ contains at most one 1:

$$\sum_{i=1}^k y_{ij}^{(s)} \in \{0, 1\}, \quad \text{for all } s \in [t], j \in [m].$$

4. For any $j \in [k], r \in [m], s \in [t], \ell \in [3m], i \in [2n]$, we introduce the new **single** formal variable $\llbracket y_{jr}^{(s)} \cdot x_{\ell i} \rrbracket$ which will stand for the *product* of two other formal variables $y_{jr}^{(s)} \cdot x_{\ell i}$. For this we shall have the following axioms:

$$y_{jr}^{(s)} - \llbracket y_{jr}^{(s)} \cdot x_{\ell i} \rrbracket \in \{0, 1\} \quad \text{and} \quad x_{\ell} - \llbracket y_{jr}^{(s)} \cdot x_{\ell i} \rrbracket \in \{0, 1\}$$

and

$$y_{jr}^{(s)} + x_{\ell i} - \llbracket y_{jr}^{(s)} \cdot x_{\ell i} \rrbracket \in \{0, 1\}$$

As an abbreviation (*not* a formal variable) we define the following:

$$Q_{ijh}^{(s)} := \sum_{r=1}^m \llbracket y_{jr}^{(s)} \cdot x_{(3(r-1)+h)i} \rrbracket, \quad \text{for all } i = [2n] \text{ and } h \in \{0, 1, 2\} \text{ and } s \in [t],$$

which expresses that x_i occurs as the h th literal in the j th clause of $\overline{Y}^{(s)}$.

5. We express that all the $\bar{Y}^{(s)}$'s are *even k -tuples* (that is, that every variable x_i appears even times) by:

$$\sum_{r \in [k], h=0,1,2} Q_{irh}^{(s)} + Q_{(i+n)rh}^{(s)} \in \{0, 2, 4, \dots, k\}, \quad \text{for all } i \in [n], s \in [t].$$

We can assume that k is even, since for every even k -tuple k must be even.

6. Similarly, we encode that the $\bar{Y}^{(s)}$'s are *inconsistent* (that is, the number of negative literals in them is odd) by:

$$\sum_{\substack{r \in [k], h=0,1,2 \\ i \in [n]}} Q_{(i+n)rh}^{(s)} \in \{0, 3, 5, \dots, k-1\}.$$

7. Every clause $i \in [m]$ appears in at most d even k -tuples $\bar{Y}^{(1)}, \dots, \bar{Y}^{(t)}$. We put:

$$\sum_{j \in [k], s \in [t]} y_{ji}^{(s)} \in \{0, 1, \dots, d\}, \quad \text{for every } i \in [m].$$

This finishes the encoding of the t inconsistent even k -tuples.

Group II of formulas (containing only \bar{X}, \bar{Z}): We now turn to the formulas expressing that there are assignments \bar{Z} that satisfy more than $m - \lceil t/d \rceil$ clauses in \bar{X} as 3XORs. For every $j \in [3m], i \in [2n], \ell \in [n]$, let $\llbracket x_{ji} \cdot z_\ell \rrbracket$ be a new formal variable that stands for the product $x_{ji} \cdot z_\ell$. As in part 4 of the formula above, we include the axioms that force $\llbracket x_{ji} \cdot z_\ell \rrbracket$ to stand for $x_{ji} \cdot z_\ell$.

Let us use the following abbreviation:

$$U_j := \sum_{h=0,1,2} \left(\sum_{i=1}^n \llbracket x_{(3(j-1)+h)i} \cdot z_i \rrbracket + \sum_{i=1}^n (x_{(3(j-1)+h)(i+n)} - \llbracket x_{(3(j-1)+h)(i+n)} \cdot z_i \rrbracket) \right).$$

Then, $U_j \in \{1, 3\}$ states that the j th clause in \bar{X} is satisfied as 3XOR by \bar{Z} . Note that $x_{(3(j-1)+h)(i+n)} - \llbracket x_{(3(j-1)+h)(i+n)} \cdot z_i \rrbracket$ is a linear term that expresses the quadratic term $x_{(3(j-1)+h)(i+n)} \cdot (1 - z_i)$.

8. Let u_j be a new formal variable expressing that the j th clause in \bar{X} is satisfied as 3XOR by \bar{Z} . Hence, $U_j \in \{1, 3\}$ iff $u_j = 1$, and we encode it as:

$$U_j \in \{0, 2\} \vee (u_j = 1) \quad \text{and} \quad U_j \in \{1, 3\} \vee (u_j = 0),$$

9. There are assignments \bar{Z} that satisfy more than $m - \lceil t/d \rceil$ clauses in \bar{X} as 3XORs:

$$\sum_{j=1}^m u_j \in \{m - \lceil t/d \rceil + 1, \dots, m\}.$$

The set of formulas described in this section has no 0, 1 solution by virtue of the 3XOR principle itself (Section 4.1).

6 Short refutations for the 3XOR principle

In this section we demonstrate polynomial-size (in n) R(quad) refutations of the 3XOR principle as encoded by disjunctions of linear equations in the previous section. We sometimes give only a high level description of the derivations. We use the terminology and abbreviations in Section 5. We also use freely the ability of R(lin) (and hence R(quad)) to count. For a detailed treatment of efficient counting arguments inside R(lin) see [RT08].

Step 1: Working in R(quad), we first show that our axioms prove that \overline{Z} cannot satisfy as 3XOR all clauses of $\overline{Y}^{(s)}$, for any $s \in [t]$.

Recall from Section 5 the abbreviation

$$Q_{ijh}^{(s)} := \sum_{r=1}^m \llbracket y_{jr}^{(s)} \cdot x_{(3(r-1)+h)i} \rrbracket, \quad \text{for all } i = [2n] \text{ and } h \in \{0, 1, 2\} \text{ and } s \in [t],$$

which stands for the statement that x_i occurs as the h th literal in the j th clause of $\overline{Y}^{(s)}$ (and where x_i for $i > n$ stands for the literal $\neg x_{i-n}$). Let us use the abbreviation:

$$P_{jhs} := \sum_{i=1}^n Q_{ijh}^{(s)} \cdot z_i + \sum_{i=1}^n Q_{(i+n)jh}^{(s)} \cdot (1 - z_i).$$

Then, P_{jhs} is a quadratic sum that stands for the statement that the h th literal in clause j in $\overline{Y}^{(s)}$ is true under \overline{Z} . Thus,

$$P_{j0s} + P_{j1s} + P_{j2s} \in \{1, 3\}, \quad \text{for all } j \in [k] \tag{4}$$

expresses that all the clauses in $\overline{Y}^{(s)}$ are satisfied as 3XOR under \overline{Z} .

Our goal now is to refute (4), based on our axioms. Informally, this refutation is done by counting: first count by clauses in $\overline{Y}^{(s)}$, namely, add all left hand sides of (4) together reaching an even number (in the right hand side) by virtue of k being even (recall we can assume that k is even). Then, count by literals, namely sum all values of literals in $Y^{(s)}$ under the assignment \overline{Z} , which we can prove is odd from our axioms. We now describe this refutation more formally.

Since k is even, counting by clauses in $\overline{Y}^{(s)}$, namely, adding the left hand sides of (4) gives us easily the following (with a polynomial-size R(quad) proof):

$$\sum_{j=1}^k P_{j0s} + P_{j1s} + P_{j2s} \in \{0, 2, 4, \dots, 3k\}. \tag{5}$$

Now we need to count by literals in $\overline{Y}^{(s)}$. We can abbreviate the number of occurrences in $\overline{Y}^{(s)}$ of the literal x_i , for $i \in [2n], s \in [t]$, by:

$$T_i := \sum_{\substack{j \in [k] \\ h=0,1,2}} Q_{ijh}^{(s)}.$$

Let us abbreviate by S_i the contribution of the literals x_i and $\neg x_i$ to the total sum (5). Thus

$$S_i := \sum_{\substack{j \in [k] \\ h=0,1,2}} Q_{ijh}^{(s)} \cdot z_i + \sum_{\substack{j \in [k] \\ h=0,1,2}} Q_{(i+n)jh}^{(s)} \cdot (1 - z_i).$$

It is possible to prove the following:

$$T_i \in \{0, 2, 4, \dots, k\} \vee S_i \in \{1, 3, 5, \dots, k-1\} \quad (6)$$

which states that if the number of occurrences in $\overline{Y}^{(s)}$ of the literal x_i is odd then (since by our axioms stating that every variable occurs even times, the number of occurrences of the literal $\neg x_i$ must also be odd) the contribution of x_i and $\neg x_i$ to the total sum (5) is also odd (because either $z_i = 0$ or $z_i = 1$).

By the axioms saying that the number of negative literals is odd (axiom 6) we get that:

$$\sum_{i=1}^n T_{i+n} \in \{1, 3, 5, \dots, k \cdot n - 1\}. \quad (7)$$

And from the axioms stating that each variable occurs even times in $\overline{Y}^{(s)}$ we have:

$$T_i + T_{i+n} \in \{0, 2, 4, \dots, k\}, \quad \text{for all } i \in [n]. \quad (8)$$

From (8) we obtain $\sum_{i=1}^{2n} T_i \in \{0, 2, 4, \dots, k \cdot n\}$, and from this and (7) we obtain

$$\sum_{i=1}^n T_i \in \{1, 3, 5, \dots, k \cdot n - 1\}. \quad (9)$$

Note that (6) can be interpreted as saying that if T_i is odd then so does S_i . Accordingly, one can use (6) to substitute all T_1, \dots, T_n in (9) with S_1, \dots, S_n , respectively. We thus get that the total sum in the left hand side of (5) is in $\{1, 3, 5, \dots\}$, and we obtain a contradiction with (5).

From a refutation of the collection of disjunctions (4), for any $s \in [t]$, we can actually get the negation of this collection, that is:

$$\bigvee_{j \in [k]} (P_{j0s} + P_{j1s} + P_{j2s}) \in \{0, 2\}. \quad (10)$$

This stems from the following: it is already true in resolution that if we have a size γ resolution refutation of A_1, \dots, A_l , then assuming the axioms $A_1 \vee B_1, \dots, A_l \vee B_l$, we can have a size $O(\gamma \cdot d)$ resolution derivation of $B_1 \vee \dots \vee B_l$, given that the total size of the B_i 's is d . To see this, take the resolution refutation of A_1, \dots, A_l and OR every line in this refutation with $B_1 \vee \dots \vee B_l$ (note that the resulting new axioms are actually derivable from the axioms $A_i \vee B_i$ via Weakening). Now, to get (10) from (4), we do the same, putting $P_{j0s} + P_{j1s} + P_{j2s} \in \{1, 3\}$ instead of A_j and $P_{j0s} + P_{j1s} + P_{j2s} \in \{0, 2\}$ instead of B_j , for all $j \in [k]$, noting that:

$$(P_{j0s} + P_{j1s} + P_{j2s} \in \{1, 3\}) \vee (P_{j0s} + P_{j1s} + P_{j2s} \in \{0, 2\}), \quad \text{for all } j \in [k]. \quad (11)$$

Step 2: The next step in our R(quad) refutation is showing how to obtain the final contradiction, given the collection of formulas (10), for all $s \in [t]$. This is again by counting: we know that for every truth assignment \bar{Z} , each $Y^{(1)}, \dots, Y^{(t)}$ must contribute at least one clause from \bar{X} that is unsatisfiable as 3XOR under \bar{Z} . We can view this as a mapping $g : [t] \rightarrow [m]$ from $Y^{(1)}, \dots, Y^{(t)}$ to the m clauses in \bar{X} , such that $g(i) = j$ means that $Y^{(i)}$ contributes the clause j in \bar{X} that is unsatisfiable under \bar{Z} as 3XOR. The mapping g is not 1-to-1, but d -to-1, because every clause of \bar{X} can appear at most d times in $Y^{(1)}, \dots, Y^{(s)}$. Our R(quad) refutation proceeds as follows.

By assumption we have $\sum_{i=1}^m u_i \in \{m - \lceil t/d \rceil + 1, \dots, m\}$, meaning that the number of clauses in \bar{X} that are satisfied as 3XOR under the assignment \bar{Z} is at least $m - \lceil t/d \rceil + 1$. Also, by the axioms in our formula, for all $i \in [m]$ we can prove that $u_i = 1$ implies that $U_i \in \{1, 3\}$; namely that the number of true literals in the i th clause of \bar{X} is 1 or 3.

For any $s \in [t]$, we can think of $\bar{Y}^{(s)}$ as a mapping $f^{(s)} : [k] \rightarrow [m]$ that maps the k clauses in $\bar{Y}^{(s)}$ to the clauses in \bar{X} . Then, $y_{ij}^{(s)} = 1$ means that $f^{(s)}(i) = j$. Thus, $u_i \cdot y_{ji}^{(s)} = 1$ means that the j th clause in $\bar{Y}^{(s)}$ is the i th clause in \bar{X} and that the i th clause in \bar{X} is satisfiable as 3XOR under \bar{Z} .

Now, it is possible to show that for any $s \in [t]$, $i \in [m]$ and $j \in [k]$, there is a proof of the following line:

$$\left(u_i \cdot y_{ji}^{(s)} = 0\right) \vee (P_{j0s} + P_{j1s} + P_{j2s} \in \{1, 3\}) \quad (12)$$

which states that if the i th clause in \bar{X} is satisfied as 3XOR under the assignment \bar{Z} and the j th clause in $\bar{Y}^{(s)}$ maps to the i th clause in \bar{X} , then the j th clause in $\bar{Y}^{(s)}$ is satisfied as 3XOR under \bar{Z} .

Informally, the proof of (12) is explained as follows: the term $P_{j0s} + P_{j1s} + P_{j2s}$ can be seen as the addition, denoted \mathcal{S} , of all inner products of the j th row of $\bar{Y}^{(s)}$ with the columns of \bar{X} (for each $h = 0, 1, 2$ we can consider the column of \bar{X} restricted to the $h \cdot i$ rows only ($i \in [m]$), and so a row of $\bar{Y}^{(s)}$ which is of length m can have an inner product with such a column of length m in \bar{X}). Because we assume that $y_{ij}^{(s)} = 1$, only the i th coordinate in the j th row of $\bar{Y}^{(s)}$ is 1 (and all the other entries in this row are 0, by our axioms). Thus, \mathcal{S} equals in fact a single column from \bar{X} ; and this single column is precisely U_i .

From (12) and (10) we can derive, for any $s \in [t]$ and any $i \in [m]$:

$$\bigvee_{j \in [k], i \in [m]} \left(y_{ji}^{(s)} \cdot (1 - u_i) = 1\right), \quad (13)$$

stating that for some $j \in [k]$, $i \in [m]$, the j th clause in $\bar{Y}^{(s)}$ is the i th clause in \bar{X} and the i th clause in \bar{X} is not satisfied as 3XOR under \bar{Z} .

Now, from (13) and axioms 7 in the 3XOR principle formulas, stating that $g : [t] \rightarrow [m]$ is d -to-1, we can obtain that the number of u_i 's that are true is no more than $m - \lceil t/d \rceil$, that is, $\sum_{i \in [m]} u_i \in \{0, \dots, m - \lceil t/d \rceil\}$, contradicting the axiom $\sum_{j=1}^m u_j \in \{m - \lceil t/d \rceil + 1, \dots, m\}$. The formal proofs of this in R(quad) is shown in the following lemma:

Lemma 8 *There are polynomial-size R(quad) refutations of (13) and the axioms in parts 7 and 9 in the 3XOR principle.*

Proof: First sum all axioms (7) to obtain:

$$\sum_{\substack{j \in [k], s \in [t] \\ r \in [m]}} y_{jr}^{(s)} \in \{0, 1, \dots, d \cdot m\}. \quad (14)$$

From (13) we can obtain:

$$\sum_{j \in [k], r \in [m]} y_{jr}^{(s)} \cdot (1 - u_i) \in \{1, 2, \dots, k \cdot m\}, \quad \text{for every } s \in [t].$$

And by summing this for all $s \in [t]$ and $i \in [m]$, we get:

$$\begin{aligned} \sum_{i \in [m]} \sum_{\substack{j \in [k], r \in [m] \\ s \in [t]}} y_{jr}^{(s)} \cdot (1 - u_i) &= \sum_{i \in [m]} (1 - u_i) \cdot \sum_{\substack{j \in [k], r \in [m] \\ s \in [t]}} y_{jr}^{(s)} \\ &\in \{t \cdot m, t \cdot m + 1, \dots, t \cdot k \cdot m^2\}. \end{aligned} \quad (15)$$

From the axiom in part (9) in the 3XOR principle $\sum_{j=1}^m u_j \in \{m - \lceil t/d \rceil + 1, \dots, m\}$ we can obtain easily

$$\sum_{i \in [m]} (1 - u_i) \in \{0, 1, \dots, \lceil t/d \rceil - 1\}.$$

From this and (14) we get, via Lemma 9 proved below, the following:

$$\sum_{i \in [m]} (1 - u_i) \cdot \sum_{\substack{j \in [k], s \in [t] \\ r \in [m]}} y_{jr}^{(s)} \in \{0, 1, \dots, d \cdot m \cdot (\lceil t/d \rceil - 1)\}.$$

Since $d \cdot m \cdot (\lceil t/d \rceil - 1) < d \cdot m \cdot \lceil t/d \rceil \leq m \cdot t$, we obtain a contradiction with (15), which finishes the refutation. \blacksquare

It remains to prove Lemma 9, which was used in the above proof:

Lemma 9 *Let $\sum_{i \in I} x_i \in \{0, 1, \dots, n\}$ and $\sum_{j \in J} y_j \in \{0, 1, \dots, m\}$ be disjunctions of linear equations, both of size at most s . Given these two disjunctions we can prove in $\text{R}(\text{quad})$ with a polynomial-size in s proof, the following:*

$$\sum_{i \in I} x_i \cdot \sum_{j \in J} y_j \in \{0, 1, \dots, m \cdot n\}. \quad (16)$$

Proof: We can reason in a case-by-case manner as follows (see [RT08] on how to carry out informal case-analysis reasoning inside $\text{R}(\text{lin})$): assume that $\sum_{j \in J} y_j = a$, for $a \in \{0, 1, \dots, m\}$. We wish to show that $x_1 \cdot \sum_{j \in J} y_j = ax_1$. If $x_1 = 0$ then $x_1 \cdot \sum_{j \in J} y_j = 0 = ax_1$. Otherwise, $x_1 = 1$. Then, $x_1 \cdot \sum_{j \in J} y_j = \sum_{j \in J} y_j = a = ax_1$. Since we have the axiom $(x_1 = 0) \vee (x_1 = 1)$ we conclude that $x_1 \cdot \sum_{j \in J} y_j = ax_1$. In a similar way we can derive for all $i \in I$:

$$x_i \cdot \sum_{j \in J} y_j = ax_i. \quad (17)$$

And by adding (17) for all $i \in I$ we obtain:

$$\sum_{i \in I} x_i \cdot \sum_{j \in J} y_j = a \cdot \sum_{i \in I} x_i.$$

Now using the axiom $\sum_{i \in I} x_i \in \{0, 1, \dots, n\}$, we get

$$\sum_{i \in I} x_i \cdot \sum_{j \in J} y_j \in \{0, a, 2a, \dots, n \cdot a\}. \quad (18)$$

Recall that (18) was obtained under the assumption that $\sum_{j \in J} y_j = a$. This means that if we have the axiom $\sum_{j \in J} y_j \in \{0, 1, \dots, m\}$, we can obtain:

$$\sum_{i \in I} x_i \cdot \sum_{j \in J} y_j \in \{b \cdot c \mid b \in \{0, 1, \dots, n\} \text{ and } c \in \{0, 1, \dots, m\}\} = \{0, 1, \dots, n \cdot m\}.$$

■

Note that the proof of Lemma 9 would also work if instead of the sums $\sum_{i \in I} x_i$ or $\sum_{j \in J} y_j$ we have $\sum_{i \in I} b_i x_i$ or $\sum_{j \in J} c_j y_j$, for integers b_i, c_j .

7 Reduction to weak automatizability of R(lin)

Here we show that R(lin) is weakly automatizable if and only if R(quad) is weakly automatizable.

To show that R(lin) is weakly automatizable iff R(quad) is weakly automatizable we use a similar idea to Pudlák [Pud03]. Namely, we show that the *canonical pair* of R(quad) is polynomially reducible to the canonical pair of R(lin).

Definition 6 ([Raz94]) *The canonical pair of a refutation system \mathcal{P} is the disjoint **NP**-pair, whose first **NP** language consists of all pairs $(\tau, 1^m)$ where τ is an unsatisfiable formula that has a \mathcal{P} -refutation of size at most m , and whose second **NP** language is the set of pairs $(\mu, 1^m)$ where μ is a satisfiable formula and m is some natural number.*

We say that a canonical pair (A, B) of a refutation system \mathcal{P}' is *polynomially reducible* to the canonical pair (A', B') of another refutation system \mathcal{P} if there is a polynomial-time computable function f such that for all x it holds that $x \in A \iff f(x) \in A'$ and $x \in B \iff f(x) \in B'$. A simple corollary of the above definitions is the following:

Proposition 10 ([Pud03]) *If the canonical pair of \mathcal{P}' is polynomially reducible to the canonical pair of \mathcal{P} then \mathcal{P}' is weakly automatizable if \mathcal{P} is weakly automatizable.*

In view of this proposition, and since R(quad) clearly polynomially simulates R(lin) (as an extension of it), it remains to show the following:

Proposition 11 *The canonical pair of R(quad) is polynomially reducible to the canonical pair of R(lin).*

Proof:(Sketch) Similar to [Pud03], the idea is to encode a product of any two variables $x_i \cdot x_j$ as a new single formal variable x_{ij} . Thus, the reduction sends all pairs $(\tau, 1^m)$ to the pair $(\tau', 1^{\text{poly}(m)})$, where τ' is obtained from τ by adding the axioms that force all new variables x_{ij} to encode the product $x_i \cdot x_j$, as shown in Section 5. ■

Corollary 12 *R(quad) is weakly automatizable iff R(lin) is weakly automatizable.*

Since R(quad) admits polynomial-size refutations of the 3XOR principle, and since weak automatizability entails feasible interpolation, we get a reduction of the problem of determining Feige et al. non-deterministic refutation algorithm to the problem of establishing weak automatizability of R(lin):

Corollary 13 *If R(lin) is weakly automatizable then there is a deterministic refutation algorithm for random 3CNFs with $\Omega(n^{1.4})$ clauses.*

8 Conclusions

We reduced the algorithmic problem of constructing a deterministic refutation algorithm for random 3CNFs with n variables and $\Omega(n^{1.4})$ clauses to the (nondeterministic) problem of proving the existence of a short proof of a certain tautology in somewhat weak proof systems, and also to the problem of establishing weak automatizability of R(lin).

We do not believe that there are polynomial-size refutations of the 3XOR principle in cutting planes or R(lin), since, loosely speaking, it is impossible in these proof systems to compose two mappings, an ability that seems to be inevitable for any proof system that can refute efficiently the 3XOR principle.

It should also be noted that it might be possible to use similar arguments to Atserias and Bonet [AB04] to show that R(quad) has feasible interpolation iff R(lin) is weakly automatizable.

Finally, it might be possible that the 3XOR principle is already efficiently provable in an apparently weaker proof system than R(quad), namely a proof system that operates with proof-lines that are disjunctions of *constant many* generalized quadratic equations (note that we did not assume any bound on the number of disjunctions in an R(quad) proof-line). This would reduce the problem of determining Feige et al. non-deterministic algorithm to the weak automatizability of the proof system $R^0(\text{lin})$ introduced in [RT08], for which *it is already known that the feasible interpolation holds by [RT08]* (though weak automatizability is unknown for it).

Acknowledgments

I wish to thank Jan Krajíček for useful comments related to this work and Neil Thapen for a useful related discussion.

References

- [AB04] A. Atserias and Maria Luisa Bonet. On the automatizability of resolution and related propositional proof systems. *Information and Computation*, 189:182–201, 2004. 1, 1, 8
- [AM12] Albert Atserias and Elitza Maneva. Mean-payoff games and propositional proofs. In *International Conference on Automata, Languages and Programming*, volume 6198 of *Lecture Notes in Computer Science*, pages 102–113. Springer Berlin / Heidelberg, 2012. 1.0.1
- [BDG⁺99] Maria Luisa Bonet, Carlos Domingo, Ricard Gavaldà, Alexis Maciel, and Toniann Pitassi. Non-automatizability of bounded-depth Frege proofs. In *Fourteenth Annual IEEE Conference on Computational Complexity (Atlanta, GA, 1999)*, pages 15–23. IEEE Computer Soc., Los Alamitos, CA, 1999. 2.1
- [BP98] Paul Beame and Toniann Pitassi. Propositional proof complexity: past, present, and future. *Bull. Eur. Assoc. Theor. Comput. Sci. EATCS*, (65):66–89, 1998. 2.1
- [BPR97] Maria Luisa Bonet, Toniann Pitassi, and Ran Raz. Lower bounds for cutting planes proofs with small coefficients. *The Journal of Symbolic Logic*, 62(3):708–728, 1997. 1, 1, 2.1
- [BPR00] Maria Luisa Bonet, Toniann Pitassi, and Ran Raz. On interpolation and automatization for Frege systems. *SIAM J. Comput.*, 29(6):1939–1967, 2000. 1, 2.1
- [CCT87] W. Cook, C. R. Coullard, and G. Turan. On the complexity of cutting plane proofs. *Discrete Applied Mathematics*, 18:25–38, 1987. 1
- [CEI96] Matthew Clegg, Jeffery Edmonds, and Russell Impagliazzo. Using the Groebner basis algorithm to find proofs of unsatisfiability. In *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing (Philadelphia, PA, 1996)*, pages 174–183, New York, 1996. ACM. 2.1
- [CK02] Peter Clote and Evangelos Kranakis. *Boolean functions and computation models*. Texts in Theoretical Computer Science. An EATCS Series. Springer-Verlag, Berlin, 2002. 2.1
- [Fei02] Uriel Feige. Relations between average case complexity and approximation complexity. In *STOC*, pages 534–543, 2002. 4.1
- [Fei07] Uriel Feige. Refuting smoothed 3CNF formulas. In *Proceedings of the IEEE 48th Annual Symposium on Foundations of Computer Science*, pages 407–417. IEEE Computer Society, 2007. 1
- [FGK05] Joel Friedman, Andreas Goerdt, and Michael Krivelevich. Recognizing more unsatisfiable random k -SAT instances efficiently. *SIAM J. Comput.*, 35(2):408–430, 2005. 1

- [FKO06] Uriel Feige, Jeong Han Kim, and Eran Ofek. Witnesses for non-satisfiability of dense random 3CNF formulas. In *Proceedings of the IEEE 47th Annual Symposium on Foundations of Computer Science*, 2006. (document), 1, 1.0.1, 3, 3, 5, 4.1, 6, 4.1
- [FO07] Uriel Feige and Eran Ofek. Easily refutable subformulas of large random 3CNF formulas. *Theory of Computing*, 3(1):25–43, 2007. (document), 1, 4.1
- [GK01] A. Goerdt and M. Krivelevich. Efficient recognition of random unsatisfiable k -SAT instances by spectral methods. In *Annual Symposium on Theoretical Aspects of Computer Science*, pages 294–304, 2001. 1
- [GL03] Andreas Goerdt and André Lanka. Recognizing more random unsatisfiable 3-SAT instances efficiently. *Electronic Notes in Discrete Mathematics*, 16:21–46, 2003. 1
- [HC99] Armin Haken and Stephen Cook. An exponential lower bound for the size of monotone real circuits. *Journal of Computer and System Sciences*, 58:326–335, 1999. 2.1
- [HP11] Lei Huang and Toniann Pitassi. Automatizability and simple stochastic games. In *ICALP (1)*, pages 605–617, 2011. 1.0.1
- [IPU94] Russel Impagliazzo, Toniann Pitassi, and Alasdair Urquhart. Upper and lower bounds for tree-like cutting planes proofs. In *Ninth Annual Symposium on Logic in Computer Science*, pages 220–228. IEEE Comput. Soc. Press, 1994. 2.1
- [Kra94] Jan Krajíček. Lower bounds to the size of constant-depth propositional proofs. *The Journal of Symbolic Logic*, 59(1):73–86, 1994. 1
- [Kra95] Jan Krajíček. *Bounded arithmetic, propositional logic, and complexity theory*, volume 60 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, 1995. 2.1
- [Kra97] Jan Krajíček. Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic. *The Journal of Symbolic Logic*, 62(2):457–486, 1997. 1, 1, 2.1
- [Kra01] Jan Krajíček. On the weak pigeonhole principle. *Fund. Math.*, 170(1-2):123–140, 2001. 1
- [MT12] Sebastian Müller and Iddo Zameret. Short propositional refutations for dense random 3CNF formulas. In *Proceedings of the 27th Annual ACM/IEEE Symposium on Logic In Computer Science (LICS)*, 2012. 1, 1, 4.1
- [Pud97] Pavel Pudlák. Lower bounds for resolution and cutting plane proofs and monotone computations. *The Journal of Symbolic Logic*, 62(3):981–998, Sept. 1997. 1, 2.1
- [Pud03] Pavel Pudlák. On reducibility and symmetry of disjoint NP pairs. *Theoret. Comput. Sci.*, 295:323–339, 2003. 1, 1, 1.0.1, 7, 10, 7

- [Raz94] Alexander A. Razborov. On provably disjoint np-pairs. *Electronic Colloquium on Computational Complexity (ECCC)*, 1(6), 1994. 6
- [Raz95] Alexander A. Razborov. Unprovability of lower bounds on circuit size in certain fragments of bounded arithmetic. *Izv. Ross. Akad. Nauk Ser. Mat.*, 59(1):201–224, 1995. 1
- [RT08] Ran Raz and Iddo Tzameret. Resolution over linear equations and multilinear proofs. *Ann. Pure Appl. Logic*, 155(3):194–224, 2008. (document), 1, 1.0.1, 5.1, 6, 6, 8
- [Seg07] Nathan Segerlind. The complexity of propositional proofs. *Bull. Symbolic Logic*, 13(4):417–481, 2007. 2.1
- [Tse68] Grigori Tseitin. *On the complexity of derivations in propositional calculus*. Studies in constructive mathematics and mathematical logic Part II. Consultants Bureau, New-York-London, 1968. 1.0.1